Toward a Better Understanding of Deep Neural Network Based Acoustic Modelling: An Empirical Investigation

Xingfu Wang, Lin Wang*, Jing Chen and Litao Wu
University of Science and Technology of China, Hefei 230027, P.R. China.
wangxfu@ustc.edu.cn, ustcwanglin@hotmail.com, {ccjj, litao3rd}@mail.ustc.edu.cn

Abstract
Recently, deep neural networks (DNNs) have outperformed traditional acoustic models on a variety of speech recognition benchmarks. However, due to system differences across research groups, although a tremendous breadth and depth of related work has been established, it is still not easy to assess the performance improvements of a particular architectural variant from examining the literature when building DNN acoustic models.

Our work aims to uncover which variations among baseline systems are most relevant for automatic speech recognition (ASR) performance via a series of systematic tests on the limits of the major architectural choices. By holding all the other components fixed, we are able to explore the design and training decisions without being confounded by the other influencing factors.

Our experiment results suggest that a relatively simple DNN architecture and optimization technique produces strong results. These findings, along with previous work, not only help build a better understanding towards why DNN acoustic models perform well or how they might be improved, but also help establish a set of best practices for new speech corpora and language understanding task variants.

1 Introduction
DEEP neural network (DNN) acoustic models have driven tremendous improvements in large vocabulary continuous speech recognition (LVCSR) in recent years (Hinton, Osindero, and Teh 2006; Vincent et al. 2010; Dahl et al. 2011; Jaitly et al. 2012). However, the underlying principles for understanding why DNNs work so much better remain not well understood. Recent research on DNN acoustic models for LVCSR have explored a great many variations different in network architecture, optimization techniques, loss functions and regularization methods to reduce overfitting.

Due to system differences across research groups it can be difficult, for example, to determine whether a performance improvement is due to a better neural network architecture or a different optimization technique. Our work aims to address this concern by systematically exploring the major strategies to improve DNN acoustic models.

This paper offers an empirical investigation of DNN performance on two LVCSR tasks to understand best practices and important design decisions when building DNN acoustic models.

The remainder of the paper is organized as follows: Section 2 reviews the previous work involved in building neural network acoustic models for LVCSR, and contextualizes the questions addressed by our investigations. Section 3 describes the neural network architectures and optimization algorithms evaluated in this paper. Section 4 presents our experiments on the Switchboard corpus, which focus on regularization choices. We then present experiments on the larger Fisher corpora in Section 5 which explore the performance of larger and deeper DNN architectures. We conclude in Section 6.

2 Related Work
To better contextualize previous work, and further convey what parts of the process are not yet fully understood, we break the discussion into the following 4 aspects of modeling and algorithmic choices:

2.1 Network Size and Depth
The total number of parameters used in modern DNNs is typically 10 to 100 times greater than neural networks used in the original hybrid HMM experiments. This increased model size, which translates to increased representational capacity, is critical to the success of modern DNN-HMM system. Using DNNs with many hidden layers and many total parameters has generally found to be beneficial (Seide, Li, and Yu 2011; Morgan 2012; Yu et al. 2013), but the role of hidden layers and total network size is not generally understood. In the work of Rich Caruana (Ba and Caruana 2013), it is stated that trained deep nets can be duplicated with shallow nets, but perhaps the deep structure is beneficial to learning. Whether deeper is always better, or how deep a network must be to obtain good performance, is not well understood both for speech recognition and DNN classification tasks more generally. We build DNNs with 5 to 10 times the total number of free parameters of DNNs used in most previous work to investigate the question of how far
we can push DNN model size or depth to continue increasing LVCSR performance.

2.2 The Loss Function

The default choice for DNN acoustic models is the cross entropy loss function, but it ignores the DNN as a component of the larger ASR system. To account for more aspects of the overall system, discriminative loss functions were introduced for ASR tasks(Kingsbury, Sainath, and Soltau 2012; Vešelý et al. 2013; Su et al. 2013). We can view discriminative training as a task-specific loss function which produces a DNN acoustic model to better act as a sub-component of the overall ASR system.

The cross entropy loss function does not consider each utterance in its entirety. Instead it is defined over individual samples of acoustic input x and senone label y. The cross entropy objective function for a single training pair \((x, y)\) is,

\[
- \sum_{k=1}^{K} 1\{y = k\} \log \hat{y}_k;
\]

Where \(K\) is the number of output classes, and \(\hat{y}_k\) is the probability that the model assigns to the input example taking on label \(k\). Cross entropy is a convex approximation to the ideal 0-1 loss for classification.

2.3 Optimization Algorithm

While stochastic gradient descent (SGD) provides a robust default choice for optimizing DNNs, recent work has shown that more advanced quasi-Newton methods can yield better results for DNN tasks generally (Martens 2010; Ngiam et al. 2011) as well as DNN acoustic modeling(Kingsbury, Sainath, and Soltau 2012). Quasi-Newton and similar methods tend to be more computationally expensive per update than SGD methods, but the improved optimization performance can sometimes be distributed across multiple processors more easily, or necessary for loss functions which are difficult to optimize well with SGD techniques.

Recently algorithms like AdaGrad (Duchi, Hazan, and Singer 2011) and Nesterovs Accelerated Gradient (NAG) were applied to DNNs for tasks outside of speech recognition, and tend to provide superior optimization as compared to SGD while still being computationally inexpensive compared to traditional quasi-Newton methods(Sutskever et al. 2013).

2.4 The Regularization Technique

Regularization is especially important for DNNs where we can easily increase models representational capacity. The simplest form of regularization widely applied to DNNs is a weight norm penalty, most often used with an l2-norm penalty.

Dropout regularization(Hinton et al. 2012) was recently introduced as a more effective regularization technique for DNN training. Several experiments demonstrate dropout as a good regularization technique for tasks in computer vision and natural language processing (Krizhevsky, Sutskever, and Hinton 2012; Wager, Wang, and Liang 2013). (Dahl, Sainath, and Hinton 2013) found a reduction in WER when using dropout on a 10M parameter DNN acoustic model for a 50 hour broadcast news LVCSR task.

The modifications to improve the generalization performance of large DNNs evaluated in our experiments include dropout, as well as early stopping, which has been used in neural network training for many years.

3 Neural Network Computations

We describe briefly here the specifics of the architecture, along with the loss function and optimization algorithms we use.

3.1 Cross Entropy Loss Function

All of our experiments utilize the cross entropy classification loss function. We choose to focus only on cross entropy because training with cross entropy is almost always the first step, or an additional loss function criterion, when experimenting with more task-specific loss functions. Additionally, the cross entropy loss function is a standard choice for classification tasks, and using it allows our experiments to serve as a case study for large scale DNN classification tasks more generally.

However, when training acoustic models perfect classification at the level of short acoustic spans is not our ultimate goal. Instead, we wish to minimize the WER of the final LVCSR system.

WER measures mistakes at the word level, and it is possible to perfectly transcribe the words in an utterance without perfectly classifying the HMM state present at each time step. Constraints present in the HMM and word sequence probabilities from the language model can correct minor errors in state-level HMM observation estimates. Conversely, not all acoustic spans are of equal importance in obtaining the correct word-level transcription. The relationship between classification accuracy rate at the frame level and overall system word error rate (WER) is complex and not well understood. In our experiments we always report both frame-level error metrics and system-level WER to eliciting insights about the relationship between DNN loss function performance and overall system performance.

3.2 Hidden Layer Computation

Traditional approaches to neural networks typically use a sigmoidal function. However, in this work we use rectified linear units which were recently shown to lead to better performance in hybrid speech recognition as well as other DNN classification tasks(Zeiler et al. 2013; Maas, Hannun, and Ng 2013).

The rectifier nonlinearity is defined as,

\[
\sigma(z) = \max(z, 0) = \begin{cases} 
  z, & z_i > 0 \\
  0, & z_i \leq 0 
\end{cases} \quad (2)
\]

The choice of rectifier nonlinearities is a new one, but their benefit has been reproduced by several research groups.
3.3 Optimization Algorithms

We consider two of the most popular stochastic gradient techniques for our neural network training. The first one is stochastic gradient with classical momentum (CM), which is the most standard choice in modern neural network research. To minimize a cost function \( f(\theta) \) classical momentum updates amount to,

\[
\begin{align*}
\nu_t &= \mu \nu_{t-1} - \epsilon \nabla f(\theta_{t-1}) \\
\theta_t &= \theta_{t-1} + \nu_t
\end{align*}
\]  

(3)

where \( \nu_t \) denotes the accumulated gradient update, or velocity, \( \epsilon > 0 \) is the learning rate, and the momentum constant \( \mu \in [0,1] \) governs how we accumulate the velocity vector over time.

By setting \( \mu \) close to one, one can expect to accumulate the gradient information across a larger set of past updates. However, it can be shown that for extremely ill-conditioned problems, a high momentum for classical momentum method might actually cause fluctuations in the parameter updates. This in turn can result in slower convergence.

Recently the Nesterov accelerated gradient (NAG) (Nesterov 1983) technique was found to address some of the issues encountered when training neural networks with CM. Both methods follow the intuition that accumulating the gradient updates along the course of optimization will help speed up convergence. NAG accumulates past gradients using an alternative update equation that finds a better objective function value with less sensitivity to optimization algorithm hyper-parameters on some neural network tasks, which is defined as,

\[
\begin{align*}
\nu_t &= \mu \nu_{t-1} - \epsilon t_{-1} \nabla f(\theta_{t-1} + \mu t_{-1} \nu_{t-1}) \\
\theta_t &= \theta_{t-1} + \nu_t
\end{align*}
\]  

(4)

Intuitively, this method avoids potential fluctuation in the optimization by looking ahead to the gradient along the update direction. For a more detailed explanation of the intuition underlying NAG optimization for neural network tasks see Figure 7.1 in (Sutskever 2013).

4 33 Hour Switchboard Corpus

We first carry out LVCSR experiments on an 33 hour sub-set of the Switchboard conversational telephone speech corpus (LDC97S62). The baseline GMM system and forced alignments are created using the Kaldi open-source toolkit\(^1\) (Povey et al. 2011).

The baseline recognizer has 3,986 sub-phone states and 20k Gaussians. Input features for the DNNs are MFCCs with a context of \( \pm 10 \) frames. Per-speaker CMVN is applied and speaker adaptation is done using fMLLR. The features are also globally normalized prior to training the DNN.

For evaluation, we report on a test set consisting of both the Switchboard and CallHome subsets of the HUB5 2000 data (LDC2002S09) as well as a subset of the training set consisting of 5,000 utterances.

4.1 Varying DNN Model Size

We explore three different model sizes by varying the number of hidden units in each layer while the number of hidden layers is fixed to 5 and all hidden layers in a single network have the same number of hidden units. The hidden layer sizes are 648, 1250 and 2992 which respectively yield models with approximately 3.6 million (M), 10M and 50M parameters.

There are 3,986 output classes which results in the output layer being the largest single layer in all our 3 networks. In DNNs of the size typically studied in the literature this output layer often consumes a majority of the total parameters in the network. For example in our 3.6M parameter model the output layer comprises 72% of all parameters. In contrast, the output layer in our 50M model is only 24% of total parameters.

For larger models we experiment with the standard input of \( \pm 10 \) context frames and additionally models trained with \( \pm 20 \) context frames.

For optimization, we use Nesterovs accelerated gradient with a smooth initial momentum schedule which we clamp to a maximum of 0.95 (Sutskever et al. 2013). The stochastic updates are on mini-batches of 512 examples. After each epoch, or full pass through the data, we anneal the learning rate by half. Training is stopped after improvement in the cross entropy objective evaluated on held out development set falls below a small tolerance threshold.

We train models for this paper in a model-parallel fashion by distributing the parameters across 4 GPUs using the distributed neural network infrastructure proposed by (Coates et al. 2013). A single pass through the training set for a 50M parameter DNN takes approximately 3.5 hours.

Results: Table 1 shows results for DNN systems in terms of frame-wise error metrics on the development set as well as word error rates on the training set and HUB5 2000 evaluation sets. The HUB5 set (EV) contains the Switchboard (SWBD) and Callhome (CH) evaluation subsets. Frame-wise error metrics were evaluated on 1.7m frames held out from the training set.

We find that substantially increasing DNN size shows clear improvements in frame-level metrics. Our 50M parameter DNN halves the development set cross entropy cost of the smaller 3.6M parameter DNN – a substantial reduction. For each increase in DNN model size there is approximately a 10% absolute increase in frame classification accuracy.

Frame-level metrics are further improved by using larger context windows. In all cases a model trained with larger context window outperforms its smaller context counterpart. Our best overall model in terms of frame-level metrics is a 50M parameter DNN with context window of 20 frames.

However, frame-level performance is not always a good proxy for WER performance of a final system. Large DNN acoustic models substantially reduce WER on the training set. Indeed, our results suggest that further training set WER reductions are possible by continuing to increase DNN model size. However, the gains observed on the training set in WER do not translate to the evaluation sets.

While there is a small benefit of using models larger than

\(^{1}\)http://kaldi.sf.net
Table 1: Results for DNNs of varying size and varying amounts of input context

<table>
<thead>
<tr>
<th>Model Size</th>
<th>Layer Size</th>
<th>Context</th>
<th>Dev CrossEnt</th>
<th>Dev Acc(%)</th>
<th>Train WER</th>
<th>SWBD WER</th>
<th>CH WER</th>
<th>EV WER</th>
</tr>
</thead>
<tbody>
<tr>
<td>GMM</td>
<td>—</td>
<td>0</td>
<td>—</td>
<td>—</td>
<td>24.93</td>
<td>21.7</td>
<td>36.1</td>
<td>29.0</td>
</tr>
<tr>
<td>3.6M</td>
<td>648</td>
<td>±10</td>
<td>1.23</td>
<td>66.20</td>
<td>17.52</td>
<td>15.1</td>
<td>27.1</td>
<td>21.2</td>
</tr>
<tr>
<td>10M</td>
<td>1250</td>
<td>±10</td>
<td>0.77</td>
<td>78.56</td>
<td>13.66</td>
<td>14.5</td>
<td>27.0</td>
<td>20.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>±20</td>
<td>0.50</td>
<td>85.58</td>
<td>12.31</td>
<td>14.9</td>
<td>27.7</td>
<td>21.4</td>
</tr>
<tr>
<td>50M</td>
<td>2992</td>
<td>±10</td>
<td>0.51</td>
<td>86.06</td>
<td>11.56</td>
<td>15.0</td>
<td>26.8</td>
<td>20.9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>±20</td>
<td>0.26</td>
<td>93.05</td>
<td>10.09</td>
<td>15.4</td>
<td>28.5</td>
<td>22.0</td>
</tr>
</tbody>
</table>

Table 2: Test set WER performance of DNN trained with dropout

<table>
<thead>
<tr>
<th>Model</th>
<th>Dropout</th>
<th>SWBD</th>
<th>CH</th>
<th>EV</th>
</tr>
</thead>
<tbody>
<tr>
<td>GMM</td>
<td>—</td>
<td>21.7</td>
<td>36.1</td>
<td>29.0</td>
</tr>
<tr>
<td>3.6M</td>
<td>DO</td>
<td>15.1</td>
<td>27.1</td>
<td>21.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>14.7</td>
<td>26.7</td>
<td>20.8</td>
</tr>
<tr>
<td>10M</td>
<td>DO</td>
<td>14.5</td>
<td>27.0</td>
<td>20.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>14.6</td>
<td>26.3</td>
<td>20.5</td>
</tr>
<tr>
<td>50M</td>
<td>DO</td>
<td>15.0</td>
<td>26.8</td>
<td>20.9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>14.9</td>
<td>26.3</td>
<td>20.7</td>
</tr>
</tbody>
</table>

Figure 1: Train and test set WER as a function of training epoch for systems with DNN acoustic models of varying size

Discussion: Figure 1 shows WER performance for the 10M and 50M parameter DNNs after each epoch of cross entropy training. We find that training WER reduces fairly dramatically at first and then continues to decrease at a slower but still meaningful rate. In contrast, nearly all of evaluation set performance is realized within the first few epochs. Although the training error rate is substantially lower for large models, there is no gain in test set performance.

This dynamics has two important practical implications for large DNN training in speech recognition.

First, large acoustic models are not beneficial but do not exhibit a strong over-fitting effect where evaluation set performance improves for a while before becoming increasingly worse.

Second, it may be possible to utilize large DNNs without prohibitively long training times by utilizing our finding that most performance comes from the first few epochs, even with models at our scale.

Finally, although increasing context window size improves all training set metrics, those gains do not translate to improved test set performance. It seems that increasing context window size provides an easy path to better fitting the training function, but does not result in the DNN learning a meaningful, generalizable function.

4.2 Dropout Regularization

The dropout technique randomly masks out hidden unit activations during training, which prevents co-adaptation of hidden units. For each example observed during training, each unit has its activation set to zero with probability $p \in [0, 0.5]$.

While networks which employ dropout during training were found effective in a number of different studies, the authors did not perform control experiments to measure the impact of dropout alone. We directly compare a baseline DNN to a DNN of the same architecture trained with dropout. This experiment tests whether dropout regularization can mitigate the poor generalization performance of large DNNs observed in Section 4.1.

Results: Table 2 shows that DNNs trained with dropout improve over the baseline for all model sizes we evaluate. The improvement is a consistent 0.2% to 0.4% reduction in absolute WER on the test set. While beneficial, dropout seems insufficient to fully harness the representational capacity of the largest models.

4.3 Early Stopping

We evaluate early stopping as another standard DNN regularization technique which may improve the generalization performance of large DNN acoustic models. By analyzing the training and test WER curves in Figure 1 we can observe the best-case performance of an early stopping approach to improving generalization. If we select the lowest test set WER the system achieves during DNN optimization,
compare systems built on our larger corpus to those trained in Section 4. This evaluation set serves as a reference point to push. First, we use the same Hub500 (Eval2000) corpus used to extract sentences from Fisher transcripts. For maximum likelihood linear regression (fMLLR) transform estimation (SAT) is done using a single feature-space maximum likelihood linear regression (fMLLR) transform on the features. Moreover, speaker adaptation is done using a single semi-tied covariance (STC) transform on the features. Among deep models there are much smaller gains as a function of depth. Models with 5 hidden layers show a clear advantage over those with 3 hidden layers, but there is little to no gain over those with a single hidden layer. Single hidden layer models perform much worse than DNNs with 3 hidden layers at multiple total parameter counts.

We next compare performance of DNN systems while keeping total model size fixed and varying the number of hidden layers in the DNN. The optimal architecture for a neural network may change as the total number of model parameters changes. There is no a priori reason to believe that 5 hidden layers is optimal for all model sizes. Furthermore, there are no good general heuristics to select the number of hidden layers.

5.1 Optimization Algorithm Choice
We apply here the same 3.6M DNN (5 hidden layers, 648 hidden units each layer) as in Section 4.1, which is a typical size for acoustic models used for conversational speech transcription in the research literature. For both the classical momentum and Nesterovs accelerated gradient optimization techniques the two key hyper-parameters are the initial learning rate $\epsilon$ and the maximum momentum $\mu_{max}$. Table 3 shows that, in terms of frame level accuracy, the NAG optimizer narrowly outperforms the CM optimizer, but WER performance across all evaluation sets are nearly identical.

For both optimization algorithms a high value of $\mu_{max}$ is important for good performance. Note most previous work in hybrid acoustic models use CM with $\mu_{max} = 0.90$, which does not appear to be optimal in our experiments.

We also found that a larger initial learning rate was beneficial. We ran experiments using $\epsilon \geq 0.05$ but do not report results because the DNNs diverged during the optimization process. Similarly, all models trained with $\epsilon = 0.001$ had WER more than 1% absolute higher on the EV test set as compared to the same architecture trained with $\epsilon = 0.01$. We thus omit the results for models trained with $\epsilon = 0.001$ from our results table.

For the remainder of our experiments we use the NAG optimizer with $\mu_{max} = 0.99$ and $\epsilon = 0.01$. These settings achieve the best performance overall in our initial experiments, and generally we have found the NAG optimizer to be somewhat more robust than the CM optimizer in producing good parameter solutions.

5.2 Number of Hidden Layers
We next compare performance of DNN systems while keeping total model size fixed and varying the number of hidden layers. The optimal architecture for a neural network may change as the total number of model parameters changes. There is no a priori reason to believe that 5 hidden layers is optimal for all model sizes. Furthermore, there are no good general heuristics to select the number of hidden layers for a particular task. Table 4 shows performance for DNNs with 1, 3, 5, and 7 hidden layers at multiple total parameter counts.

The most striking distinction in terms of both frame classification and WER is the performance gain of deep models versus those with a single hidden layer. Single hidden layer models perform much worse than DNNs with 3 hidden layers or more. Among deep models there are much smaller gains as a function of depth. Models with 5 hidden layers show a clear gain over those with 3 hidden layers, but there is little to no gain from a 7 hidden layer model when compared with a 5-layer model.
hidden layer model. These results suggest that for this task 5 hidden layers may be deep enough to achieve good performance, but that DNN depth taken further does not increase performance.

It’s also interesting to note that DNN depth has a much larger impact on performance than total DNN size. For this task, it is much more important to select an appropriate number of hidden layers than it is to choose an appropriate total model size.

For each total model size there is a slight decrease in frame classification in 7 layer DNNs as compared with 5 hidden layer DNNs. This trend of decreasing frame-level performance is also present in the training set, which suggests that as networks become very deep it is more difficult to minimize the training objective function.

This is evidence for a potential confounding factor when building DNNs. In theory deeper DNNs should be able to model more complex functions than their shallower counterparts, but in practice we found that depth can act as a regularizer due to the difficulties in optimizing very deep models.

### 6 Conclusion

The multi-step process of building neural network acoustic models comprises a large design space with a broad range of previous work. Our work sought to answer the question of which of the most fundamental DNN design decisions are most relevant for final ASR system performance. We found that increasing model size and depth are simple but effective ways to improve WER performance, but only up to a certain point.

For the Switchboard corpus, we found that regularization can improve the performance of large DNNs which otherwise suffer from over-fitting problems. However, a much larger gain was achieved by utilizing the larger 210hr training corpus as opposed to applying regularization with less training data. When trained with the simple NAG optimization procedure, these large DNNs achieved clear gains on both frame classification and WER when the training corpus was large.

Overall, total network size, not depth, was the most critical factor we found in our experiments. Depth is certainly important with regards to having more than one hidden layer, but differences among DNNs with multiple hidden layers were fairly small with regards to all metrics we evaluated. At a certain point it appears that increasing DNN depth yields no performance gains, and may indeed start to harm performance.

When applying DNN acoustic models to new tasks it appears sufficient to use a fixed optimization algorithm, we suggest NAG, and cross-validate over total network size using a DNN of at least three hidden layers, but no more than five.

Based on our results, this procedure should instantiate a reasonably strong baseline system for further experiments, by modifying whatever components of the acoustic model building procedure researchers choose to explore.

We trained DNNs using approximately 300 lines of Python code, demonstrating the feasibility of fairly simple architectures and optimization procedures to achieve good system performance. We make our DNN training code available online³, hoping that this serves as a reference point to improve communication and reproducibility in the now highly active research area of neural networks for speech and language understanding.

### 7 Acknowledgments

The authors would like to thank the Supercomputing Center of USTC for their help in providing the GPU hardware platform for our experiments to carry on. This work was supported in part by the National Natural Science Foundation of China (No. 61272472, No. 61472382) and the National Science and Technology Major Project of China (No. 2012ZX10004301).

³https://github.com/ustcwanglin/FeedForwardDNNTrain.git
References


Jaitly, N.; Nguyen, P.; Senior, A. W.; and Vanhoucke, V. 2012. Application of pretrained deep neural networks to large vocabulary speech recognition. In INTERSPEECH.


